The representation of a point in time using an integer to store the number of seconds since January 1, 1970, is commonly known as Unix time or POSIX time. This representation typically uses a 32-bit signed integer. The problem of such a system running out of capacity is known as the Year 2038 problem or the Unix Millennium Bug.

When Will the Problem Occur?

A 32-bit signed integer can represent values from -2^{31} to $2^{31} - 1$, which is from $-2,147,483,648$ to $2,147,483,647$.

- The start of Unix time (0) corresponds to January 1, 1970.
- The maximum positive value, $2,147,483,647$ seconds from January 1, 1970, corresponds to **03:14:07 UTC on January 19, 2038**.

Therefore, programs using this 32-bit representation will face an overflow issue at **03:14:08 UTC on January 19, 2038**, when the value increments to $2,147,483,648$, which cannot be represented within a 32-bit signed integer and will cause it to wrap around to $-2,147,483,648$.

What Should Be Done to Address This?

To mitigate the Year 2038 problem, several approaches can be taken:

1. **Transition to a 64-bit Representation**:
    - Use a 64-bit signed integer to store the number of seconds since January 1, 1970. This allows for a significantly larger range of dates:
        - The minimum value: $-9,223,372,036,854,775,808$ seconds before January 1, 1970, which is far into the past.
        - The maximum value: $9,223,372,036,854,775,807$ seconds after January 1, 1970, which extends to approximately the year 292 billion AD.
2. **Use Alternative Time Representations**:
    - Some systems might adopt different time representations or abstractions that are less prone to such overflow issues. For instance, using date-time libraries that encapsulate time representation and handle edge cases internally.
3. **Update Legacy Systems and Software**:
    - Audit existing software and systems to identify and update parts that use 32-bit time representations.
    - Where feasible, replace or update old libraries and frameworks to versions that support 64-bit time representations.

Practical Steps for Transition

- **Assessment**: Conduct an assessment of systems and software that rely on 32-bit Unix time.
- **Planning**: Develop a migration plan to transition to 64-bit time representations.
- **Implementation**: Implement the changes, including updating data storage formats, APIs, and user interfaces where necessary.
- **Testing**: Rigorously test the updated systems to ensure they handle dates correctly, especially around critical boundary values like the transition period in 2038.
- **Deployment**: Deploy the updates in a phased manner to minimize disruptions.
- **Monitoring and Maintenance**: Continuously monitor the systems for any unforeseen issues and maintain them to ensure compatibility with future standards.

By taking these steps, you can ensure that your programs continue to function correctly well beyond January 19, 2038.